Introduction to Word2vec and its application to find predominant word senses

Huizhen Wang
NTU CL Lab
2014-8-21

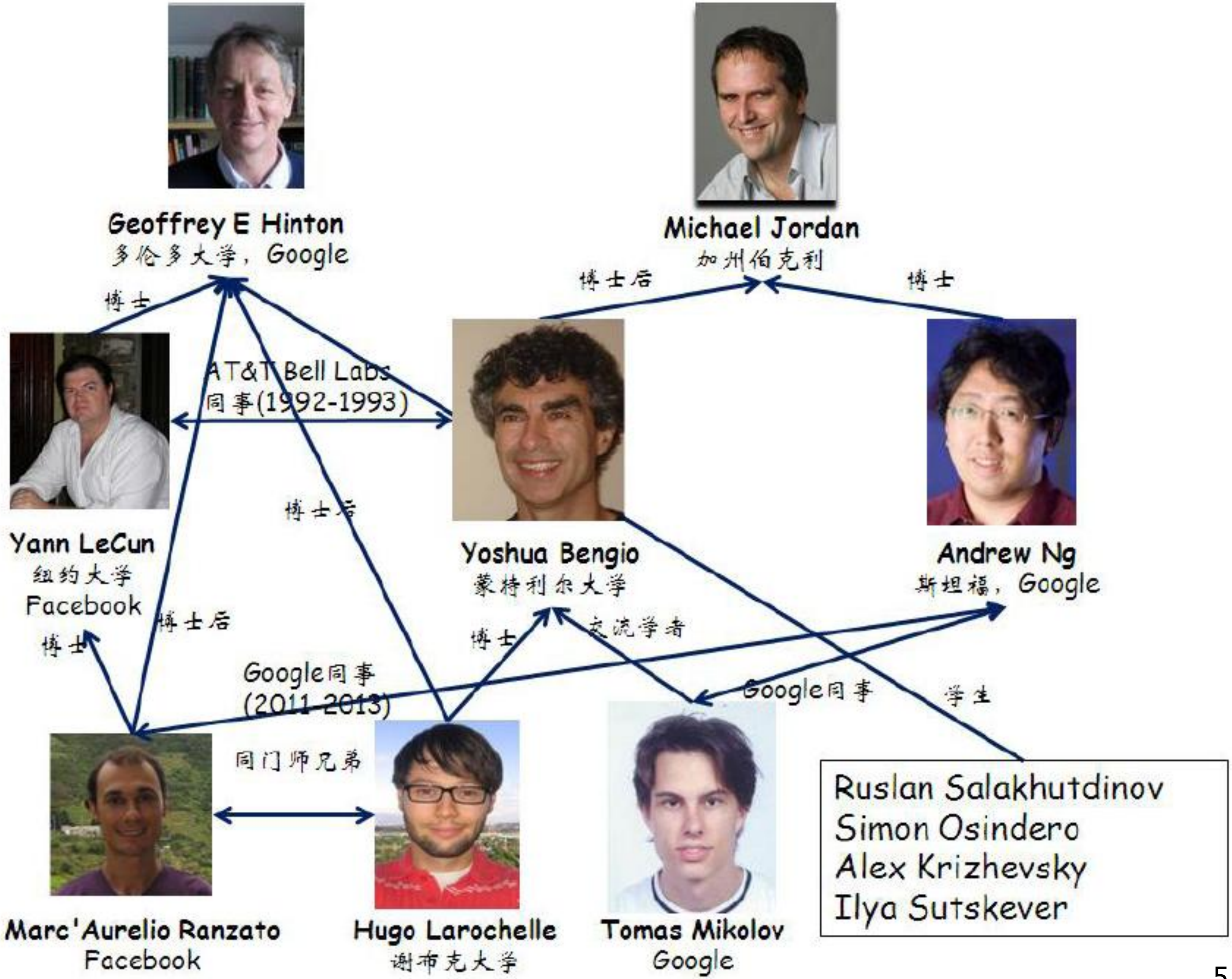
Part 1: Introduction to Word2vec

Outline

- What is word2vec?
- Quick Start and demo
- Training Model
- Applications

What is word2vec?

- Word2vec is a tool which computes vector representations of words.
- word meaning and relationships between words are encoded spatially
- learns from input texts
- Developed by Mikolov, Sutskever, Chen, Corrado and Dean in 2013 at Google Research



Geoffrey E Hinton
多伦多大学，Google
Michael Jordan
加州伯克利
博士后
博士
博士
AT&T Bell Labs
同事(1992-1993)
Yann LeCun
纽约大学
Facebook
Yoshua Bengio
蒙特利尔大学
Andrew Ng
斯坦福，Google
博士后
博士后
博士
博士
交流学者
Google同事
(2011-2013)
Google同事
学生
同门师兄弟
Ruslan Salakhutdinov
Simon Osindero
Alex Krizhevsky
Ilya Sutskever
Marc'Aurelio Ranzato
Facebook
Hugo Larochelle
谢布克大学
Tomas Mikolov
Google

Quick Start

- Download the code:
  - svn checkout http://word2vec.googlecode.com/svn/trunk/
- Run 'make' to compile word2vec tool
- Run the demo scripts: *./demo-word.sh* and *./demo-phrases.sh*

Different versions of word2vec

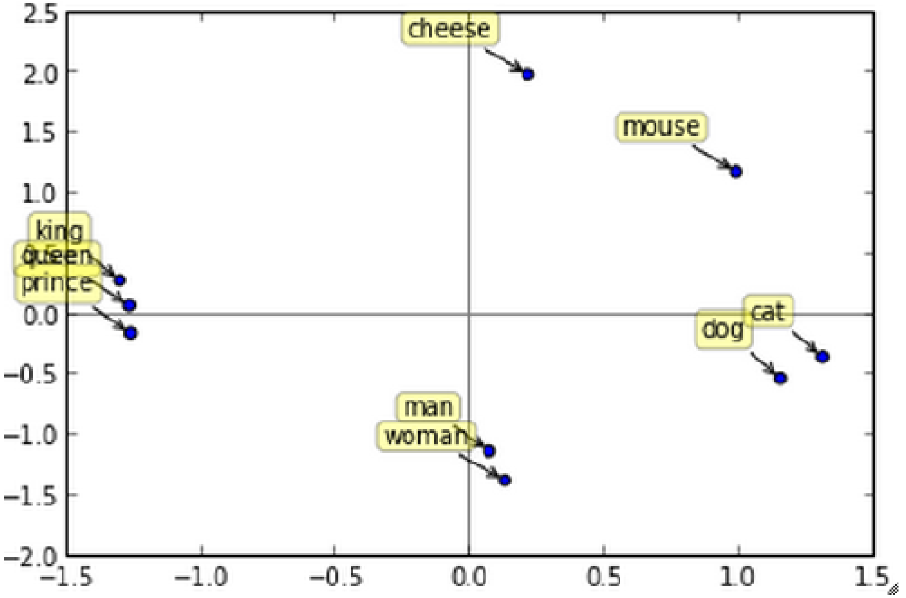
- Google code：http://word2vec.googlecode.com/svn/trunk/
- 400 lines C++11 version：https://github.com/jdeng/word2vec
- Python version: http://radimrehurek.com/gensim/models/word2vec.html
- Java ：https://github.com/ansjsun/word2vec_java
- Parallel java version：https://github.com/siegfang/word2vec
- CUDA version：https://github.com/whatupbiatch/cuda-word2vec

- Demo

- *vector('Paris') - vector('France') + vector('Italy') = ?*
- *vector('king') - vector('man') + vector('woman') = ?*

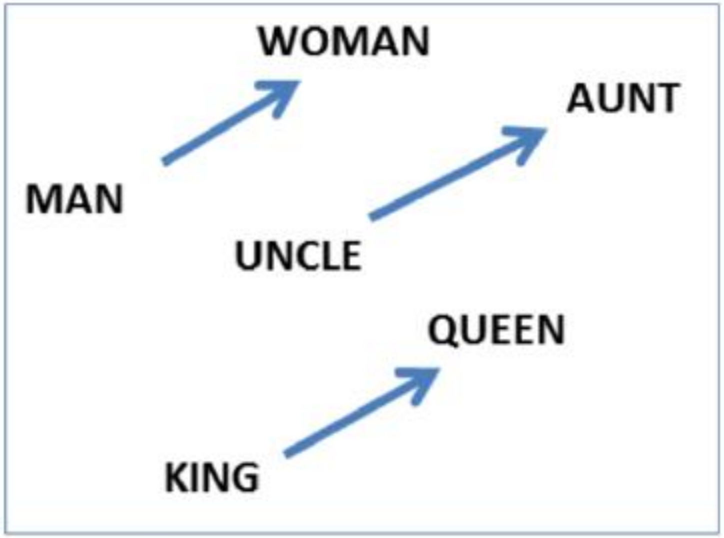
Similar words are closer together

- spatial distance corresponds to word similarity
- words are close together ⇔ their "meanings" are similar
- notation: word w -> vec[w] its point in space, as a position vector.
- e.g. vec[woman] = (0.1, -1.3)



Word relationships are displacements

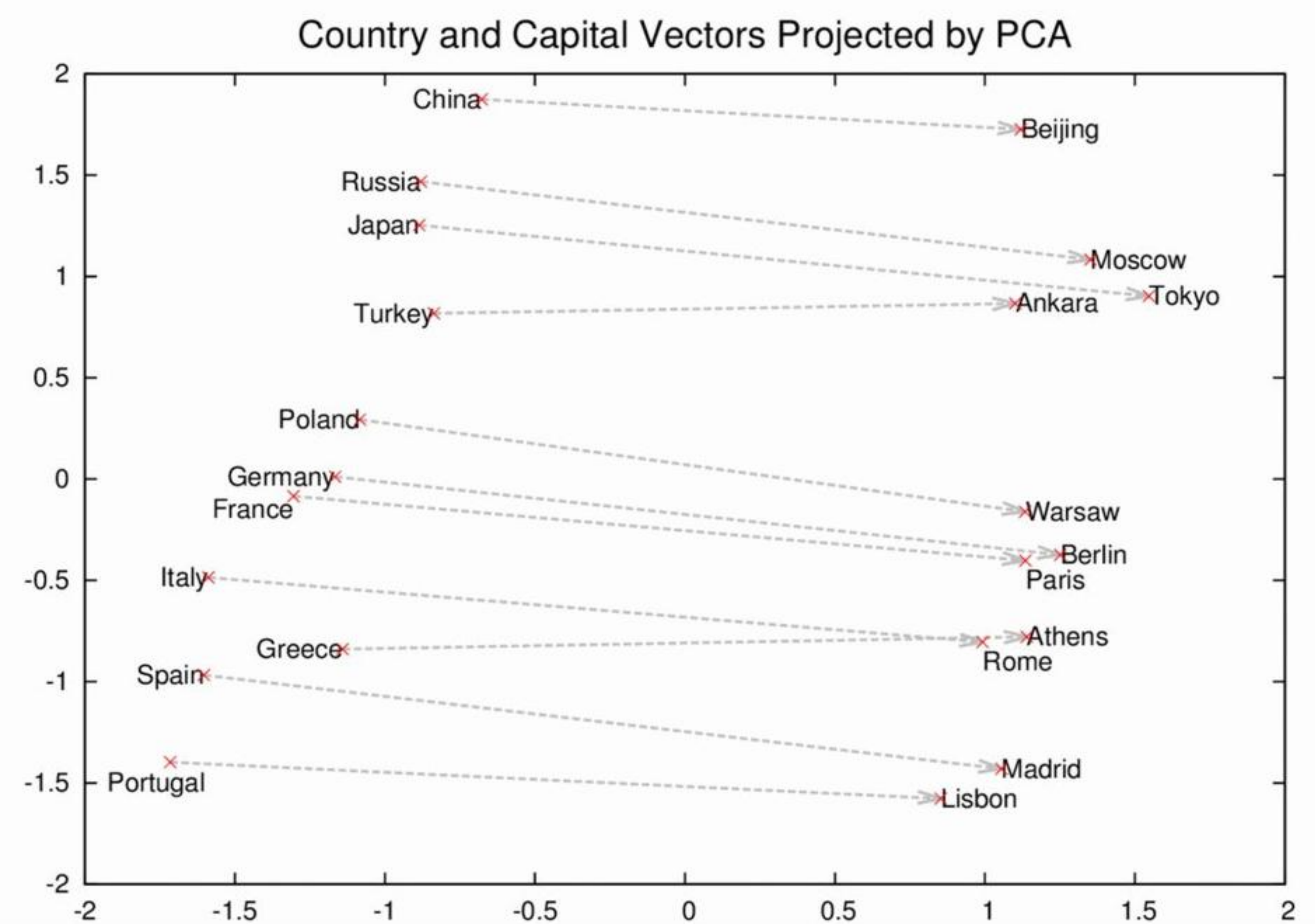
- The displacement (vector) between the points of two words represents the word relationship.
- Same word relationship => same vector



Source: *Linguistic Regularities in Continuous Space Word Representations*, Mikolov et al, 2013

- E.g. vec[queen] - vec[king] = vec[woman]- vec[man]

learn the concept of capital cities



Semantic-syntactic word relationship

Table 1: *Examples of five types of semantic and nine types of syntactic questions in the Semantic-Syntactic Word Relationship test set.*

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

Examples of the learned relationships

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

efficiency

Table 6: *Comparison of models trained using the DistBelief distributed framework. Note that training of NNLM with 1000-dimensional vectors would take too long to complete.*

Model	Vector Dimensionality	Training words	Accuracy [%]			Training time [days x CPU cores]
			Semantic	Syntactic	Total	
NNLM	100	6B	34.2	64.5	50.8	14 x 180
CBOW	1000	6B	57.3	68.9	63.7	2 x 140
Skip-gram	1000	6B	66.1	65.1	65.6	2.5 x 125

What’s in a name?

- Assume the Distributional Hypothesis (D.H.) (Harris, 1954):
  - “You shall know a word by the company it keeps” (Firth, J. R. 1957:11)

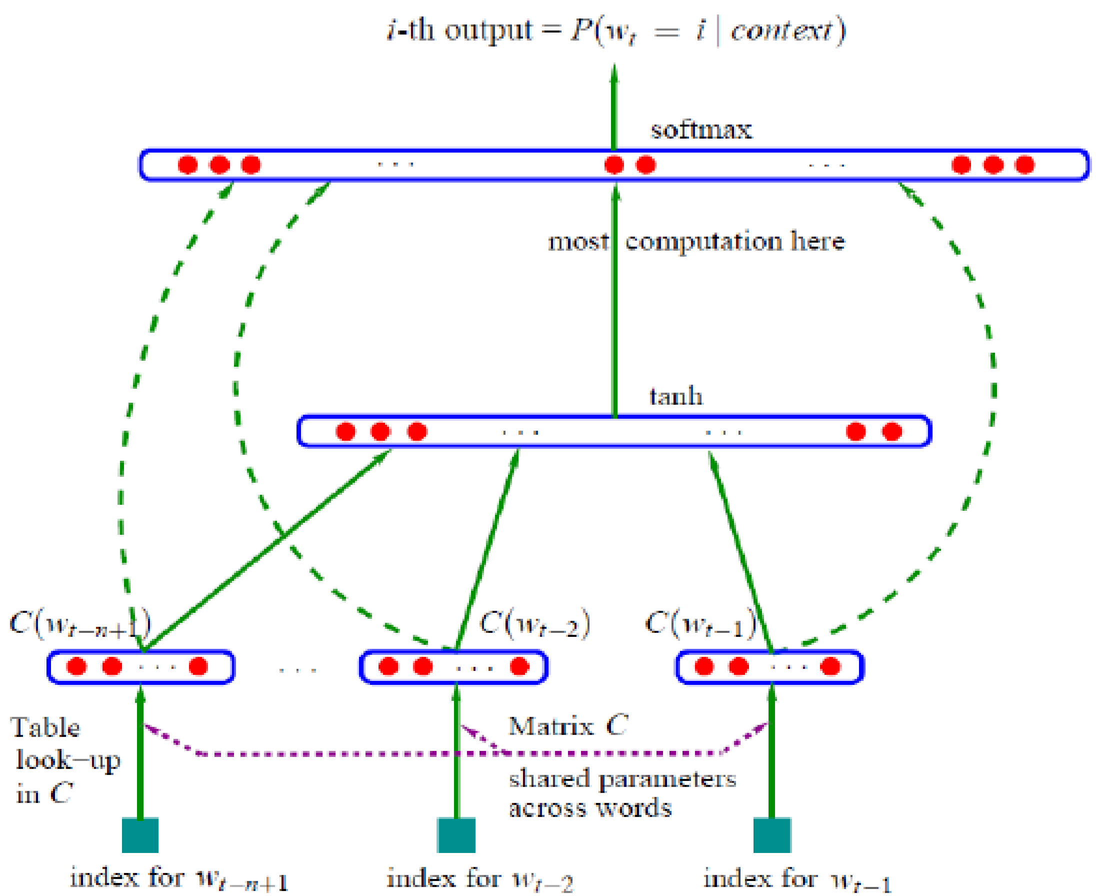
Word2vec as shallow learning

- word2vec is a successful example of “shallow” learning
- word2vec can be trained as a very simple neural network
  - single hidden layer with no non-linearities
  - no unsupervised pre-training of layers (i.e. no deep learning)
- word2vec demonstrates that, for vectorial representations of words, shallow learning can give great results.

Two approaches: CBOW and Skip-gram

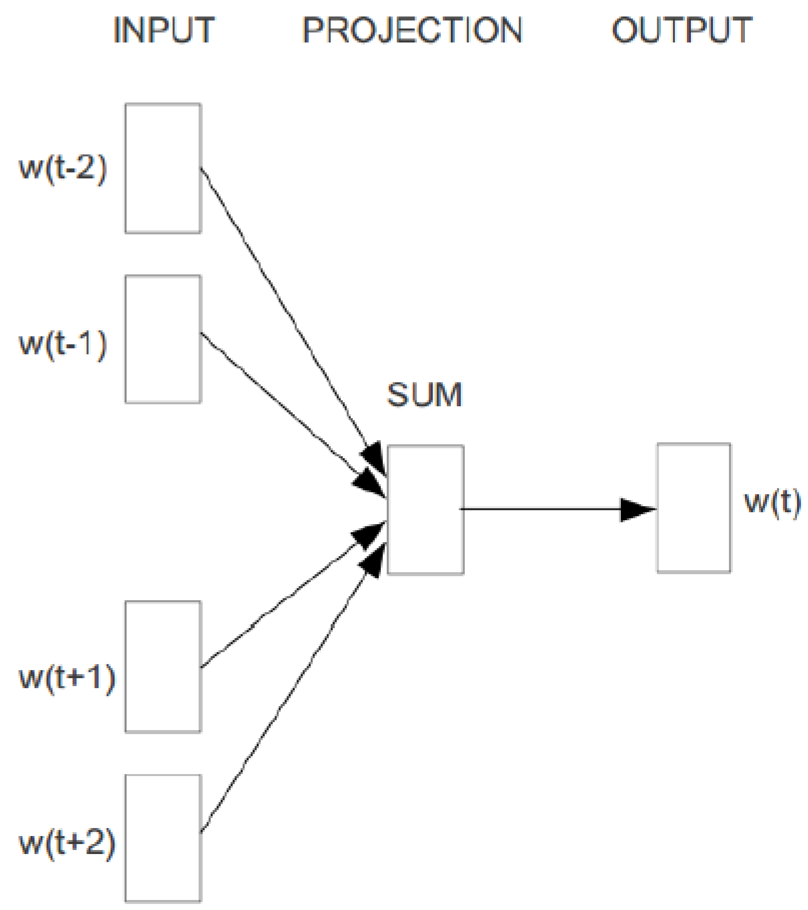
- word2vec can learn the word vectors via two distinct learning tasks, CBOW and Skip-gram.
  - CBOW: predict the current word w0 given only C
    - Hierarchical softmax
    - Negative sampling
  - Skip-gram: predict words from C given w0
    - Hierarchical softmax
    - Negative sampling
- Skip-gram produces better word vectors for infrequent words
- CBOW is faster by a factor of window size – more appropriate for larger corpora

A Neural Model (NNLM)



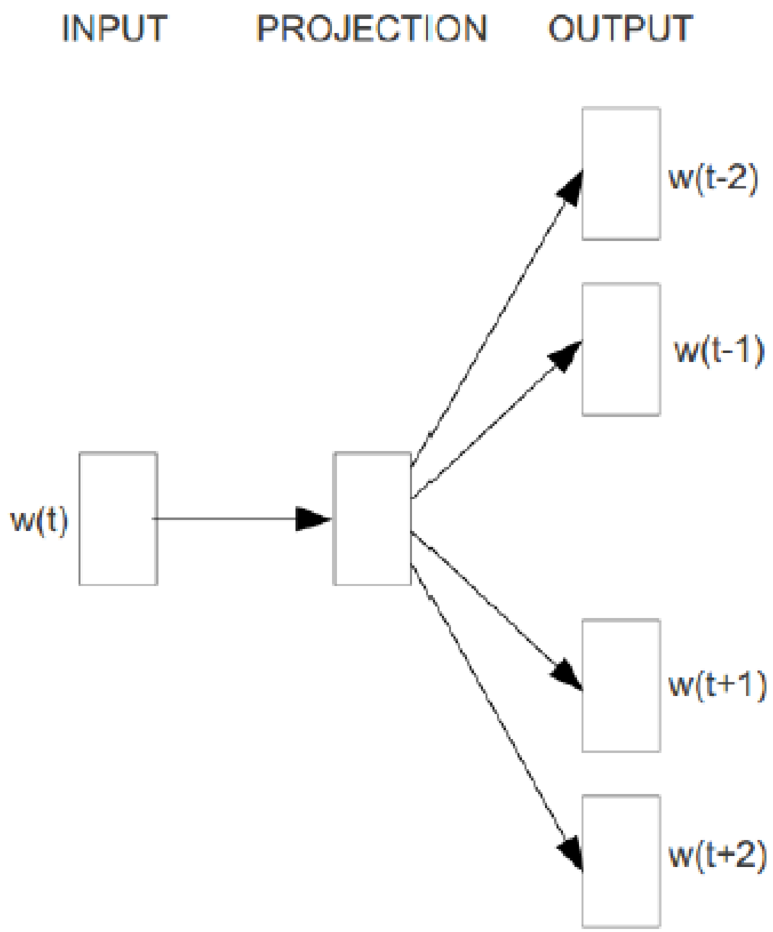
CBOW (Continuous bag of words)

- Predicting the current word based on the context
- Disregard grammar and work order
- Share the weight of each words
- Training around words



Continuous Skip-gram Model

- Maximize classification of a word based on another word in the same sentence
- The more distant words are usually less related to the current word than those close to it.



Comparison of publicly available word vectors on the Semantic-Syntactic Word Relationship test set, and word vectors from our models. Full vocabularies are used

Model	Vector Dimensionality	Training words	Accuracy [%]		
			Semantic	Syntactic	Total
Collobert-Weston NNLM	50	660M	9.3	12.3	11.0
Turian NNLM	50	37M	1.4	2.6	2.1
Turian NNLM	200	37M	1.4	2.2	1.8
Mnih NNLM	50	37M	1.8	9.1	5.8
Mnih NNLM	100	37M	3.3	13.2	8.8
Mikolov RNNLM	80	320M	4.9	18.4	12.7
Mikolov RNNLM	640	320M	8.6	36.5	24.6
Huang NNLM	50	990M	13.3	11.6	12.3
Our NNLM	20	6B	12.9	26.4	20.3
Our NNLM	50	6B	27.9	55.8	43.2
Our NNLM	100	6B	34.2	**64.5**	50.8
CBOW	300	783M	15.5	53.1	36.1
Skip-gram	300	783M	**50.0**	55.9	**53.3**

Main Parameters for training

- 1. –size： size of word vector
- 2. –window：max skip length between words
- 3. –sample：threshold for occurrence of words
- 4. –hs：using Hierarchical softmax
- 5. –negative： number of negative examples
- 6. –min-count：discard words that appear less than # times
- 7. –alpha：the starting learning rate
- 8. –cbow： using CBOW algorithm or skip-gram model

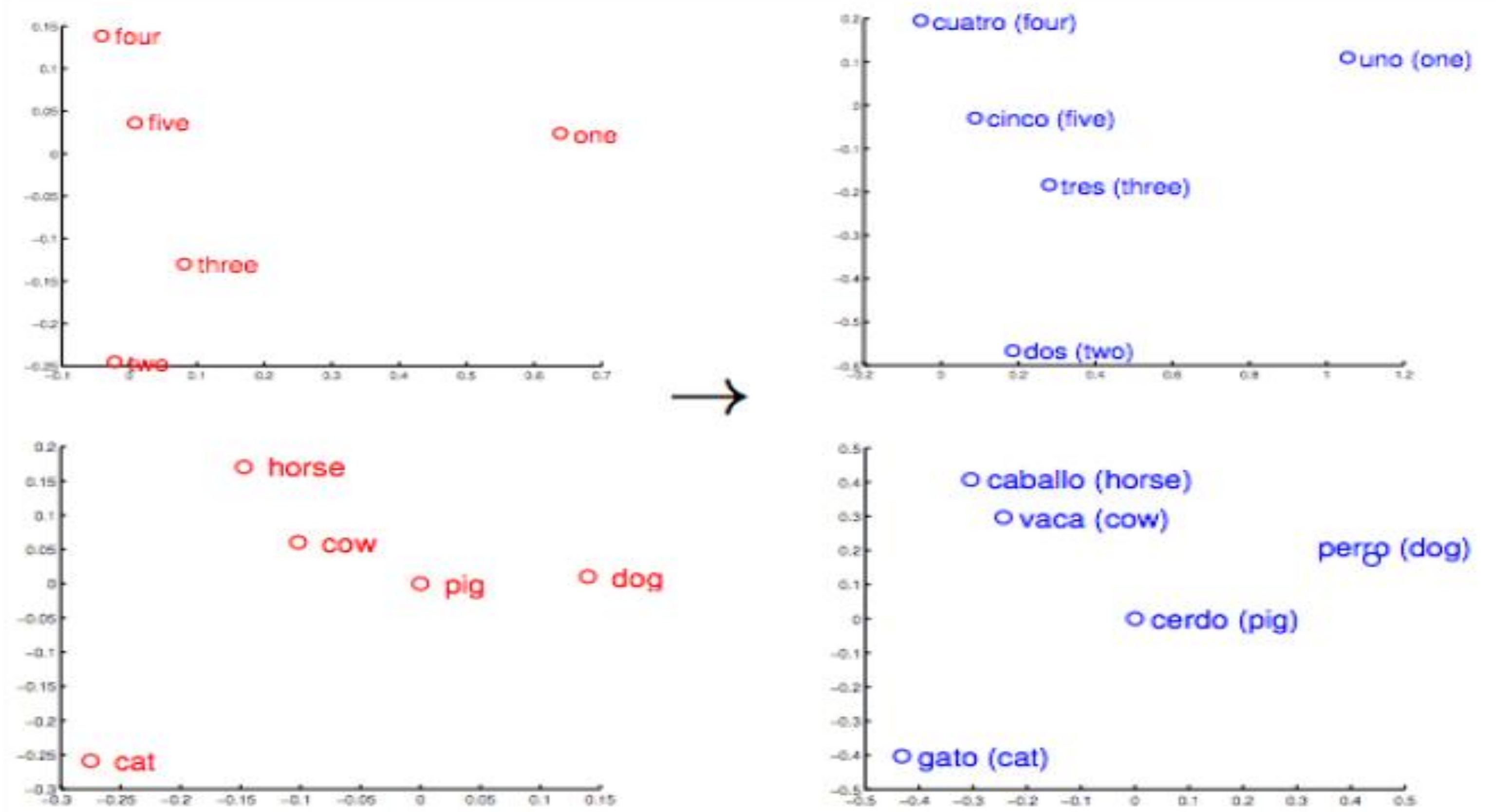
Applications

- Word segmentation
- Word cluster
- Find synonym
- Part-of-speech tagging

application to machine translation

- train word representations for e.g. English and Spanish separately
- the word vectors are similarly arranged!
- learn a linear transform that (approximately) maps the word vectors of English to the word vectors of their translations in Spanish
- same transform for all vectors

application to machine translation



Source: Exploiting Similarities among Languages for Machine Translation, Mikolov, Quoc, Sutskever, 2013

applications to machine translation - results

- English - Spanish: can guess the correct translation in 33% - 35% percent of the cases.

Translation	Edit Distance		Word Co-occurrence		Translation Matrix	
	P@1	P@5	P@1	P@5	P@1	P@5
En → Sp	13%	24%	19%	30%	33%	51%
Sp → En	18%	27%	20%	30%	35%	52%
En → Cz	5%	9%	9%	17%	27%	47%
Cz → En	7%	11%	11%	20%	23%	42%

Source: Exploiting Similarities among Languages for Machine Translation, Mikolov, Quoc, Sutskever, 2013

Reference

- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In Proceedings of Workshop at ICLR, 2013.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of NIPS, 2013.
- Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. In Proceedings of NAACL HLT, 2013.

Part 2: Finding Predominant Word Senses in Untagged text

Motivation: e.g. Dog as a noun

Noun

- (42)S: (n) **dog#1**, domestic dog#1, Canis familiaris#1 (a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds) *"the dog barked all night"*
- S: (n) frump#1, **dog#2** (a dull unattractive unpleasant girl or woman) *"she got a reputation as a frump"; "she's a real dog"*
- S: (n) **dog#3** (informal term for a man) *"you lucky dog"*
- S: (n) cad#1, bounder#1, blackguard#1, **dog#4**, hound#2, heel#3 (someone who is morally reprehensible) *"you dirty dog"*
- S: (n) frank#2, frankfurter#1, hotdog#3, hot dog#3, **dog#5**, wiener#2, wienerwurst#1, weenie#1 (a smooth-textured sausage of minced beef or pork usually smoked; often served on a bread roll)
- S: (n) pawl#1, detent#1, click#3, **dog#6** (a hinged catch that fits into a notch of a ratchet to move a wheel forward or prevent it from moving backward)
- S: (n) andiron#1, firedog#1, **dog#7**, dog-iron#1 (metal supports for logs in a fireplace) *"the andirons were too hot to touch"*

Predominant Score of word “dog_n”

- Synset(‘dog.n.01’) 24.26
- Synset(‘cad.n.01’) 17.19
- Synset(‘dog.n.03’) 17.04
- Synset(‘frump.n.01’) 16.75
- Synset(‘andiron.n.01’) 12.91
- Synset('pawl.n.01') 12.34
- Synset('frank.n.02') 7.95

Introduction

- Our work is aimed at discovering the predominant senses from raw text.
  - Hand-tagged data is not always available
  - Can produce predominant senses for the domain type required.
- We believe that automatic means of finding a predominant sense can be useful for systems that use it as backing-off and as lexical acquisition under limiting-size hand-tagges sources.

Method (McCarthy et al. 2004)

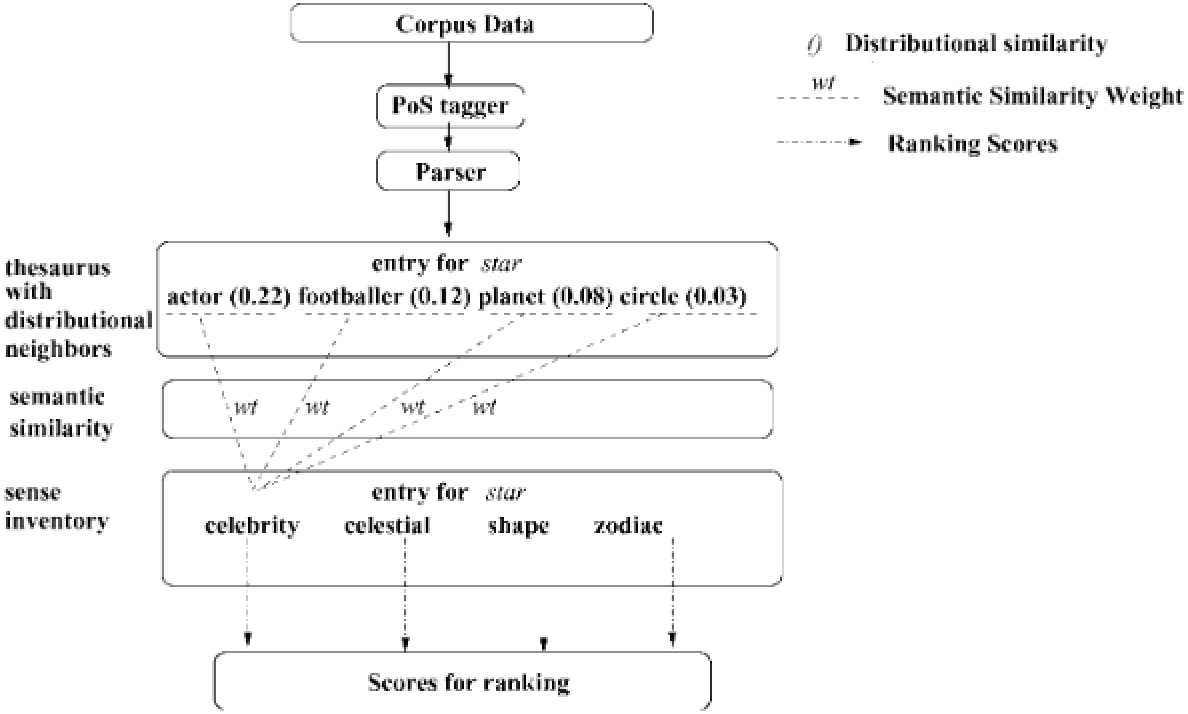


Figure 1
The prevalence ranking process for the noun *star*.

Our Method

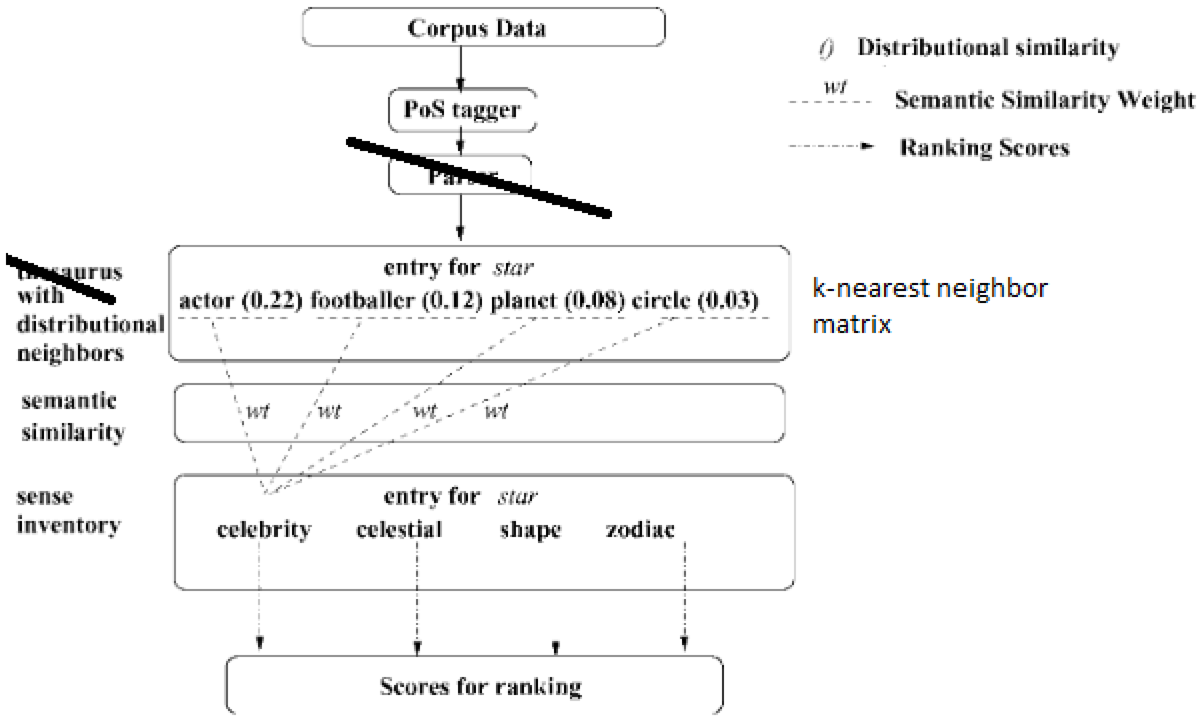


Figure 1
The prevalence ranking process for the noun *star*.

Calculation Measures

- DSS (Distributional Similarity Score)
  - K-Nearest Neighbor (k-NN)
    - Context Window Length = 3, 4, 5, 6, 7
    - Frequency as weight
  - Word2vec
- SSS (Semantic Similarity Score)
  - Wu-Palmer Similarity (wup)
  - Leacock-Chodorow Similarity (lch) (better)

Corpora Details (wikipedia dumps)

	No. of Files	No. of sentences	No. of words	No. of word types
English	19,894	85,236,022	1,747,831,592	10,232,785
Chinese	1,374	4,892,274	128,195,456	2,313,896
Japanese	3,524	11,358,127	339,897,766	1,841,236
Indonesia	514	2,168,160	38,147,344	876,288
Italian	4,143	13,225,000	355,748,901	5,805,013
Portuguese	2,232	8,339,996	192,981,797	4,464,919

Multi-Word Expression (MWE in the Wordnet)

Taylor	NNP
V.	NNP
United	NNP
States	NNPS

Taylor	NNP
V.	NNP
United States	NP

Experimental results – part of English

No. of context window	No. of Lex	Accuracy(%)
3		49.70/~51.16
4		
5		
6		
7		51.44
8		
9		
10		

Experimental results – Mandarin Chinese

No. of context window	No. of Lex	Accuracy(%)
3	1,812	67.16
4	1,813	67.18
5	1,814	68.08/~30
6	1,817	67.25
7	1,818	67.49
8	1,818	67.44
9	1,818	67.33
10	1,818	67.05

Experimental results – Indonesian

No. of context window	No. of Lex	Accuracy(%)
3	744	63.04
4	746	62.60
5	750	61.87
6	753	61.75
7	753	61.89
8	753	61.75
9	754	61.14
10	754	60.74

Conclusions

- We have devised a method that use raw corpus data to automatically find a predominant sense of nouns in WordNet.
- we investigated the effect of the frequency and choice of distributional similarity measure and apply our method for words whose PoS other than noun. –Already working with all PoS
- In the future we will look at applying to domain specific subcorpora
- Have successfully applied our processes to multiple languages (with some limitations)
  - The only sense ranking available for many languages!